# Migros Cross-Math Solver

Pascal Lüscher

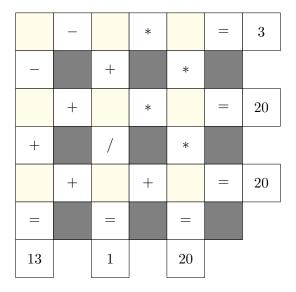
January 14, 2020

### 1 Problem

The Cross-Math puzzle is a puzzle with 9 numbers to fill. The rules are simple:

- Mathematical operation ordering does not count, simply calc from left to right and from top to bottom
- Each cell is filled with a number between 1 9
- Each number can only appear once

The puzzle presented is the following:



## 2 QCP Mathematical formulation

In this section i present some quadratic constraint programming solutions

### 2.1 With Big-M

M is very big and  $\epsilon$  is a very small number.

Equation 2 – Equation 5 are the basic requirements for the corss-math puzzle. Equation 6 – Equation 11 are the specific equations for this problem.

#### 2.2 One Hot Matrix

 $x_{i,j}$ 

The idea behind a one hot matrix is to add a binary variable for each possibility. In our case this means to add 9 binary variables for every field.

min 0 (15)  
s.t. 
$$\sum_{j=1}^{j \le 9} x_{i,j} = 1, \qquad \forall i \in \{1, \dots, 9\}$$
(16)  

$$\sum_{i \le 9} x_{i,j} = 1, \qquad \forall j \in \{1, \dots, 9\}$$
(17)  

$$y_i = \sum_{j=1}^{j \le 9} j x_{i,j}, \qquad \forall i \in \{1, \dots, 9\}$$
(18)  

$$(y_0 - y_1) y_2 = 3$$
(19)  

$$(y_3 + y_4) y_5 = 20$$
(20)  

$$y_6 + y_7 + y_8 = 20$$
(21)  

$$y_0 - y_3 + y_6 = 13$$
(22)  

$$\frac{(y_1 + y_4)}{y_7} = 1$$
(23)  

$$y_2 y_5 y_8 = 20$$
(24)  

$$x_{i,j} \in \{0, 1\}, \qquad \forall i, j \in \{1, \dots, 9\}$$
(25)

The constraints Equation 16 and Equation 17 define the uniqueness of a number in the problem. With Equation 18 a linear expression  $y_i$  is defined that expresses the actual value of the cell i. This linear expression is then used in Equation 19 – Equation 24 to define the actual problem. These constraints are the same as Equation 6 – Equation 11.

(26)

## 3 Backtracking

The puzzle is small and can easily be solved using a naive backtracking algorithm. A pseudo code algorithm is shown in Listing 3. The function checkAllPredicates checks if all conditions are met.

```
int[] field = new int[9];
Func<int,int,int,bool> predicates;
Solve(int currentField, int[] availableNumbers)
  if (!checkAllPredicates) return false;
  if (currentField > 8) return true

foreach(number in availableNumbers) do
  field[currentField] = number
   if (Solve(currentField + 1, availableNumbers without number))
    return true
  field[currentField] = 0
  return false
```

### 4 Solution

| 9  | _ | 6 | * | 1  | = | 3  |
|----|---|---|---|----|---|----|
| _  |   | + |   | *  |   |    |
| 3  | + | 2 | * | 4  | = | 20 |
| +  |   | / |   | *  |   |    |
| 7  | + | 8 | + | 5  | = | 20 |
| =  |   | = |   | =  |   |    |
| 13 |   | 1 |   | 20 |   |    |